U.S. DEPARTMENT OF ENERGY

Argonne NATIONAL LABORATORY

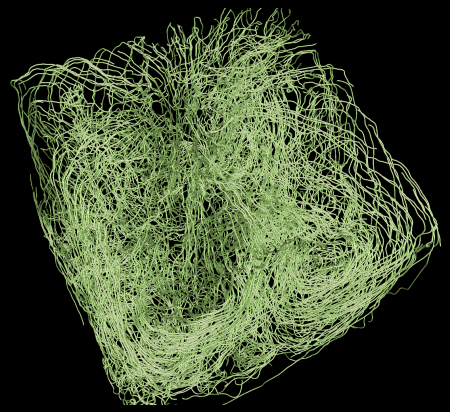
Reducing the Cost of Data Movement in Exascale Analytics

"Data movement, rather than computational processing, will be the constrained resource at exascale." – Dongarra et al. 2011.

Kitware Seminar
March 24, 2016

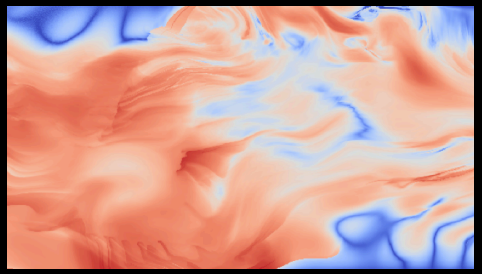
Tom Peterka
tpeterka@mcs.anl.gov
http://www.mcs.anl.gov/~tpeterka
Mathematics and Computer Science Division

Examples

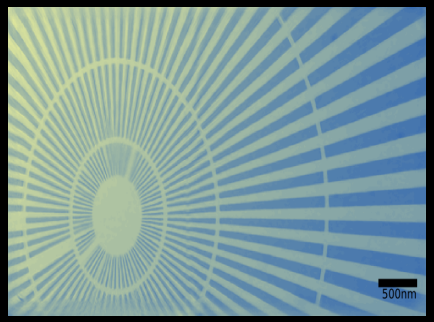


Streamlines and pathlines
in nuclear engineering

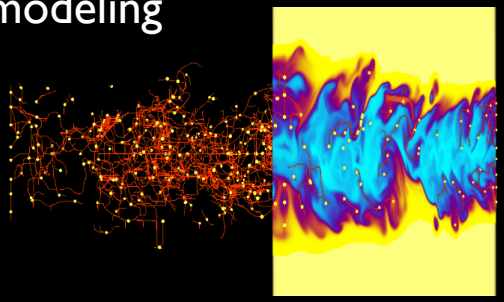
Stream surfaces
in meteorology



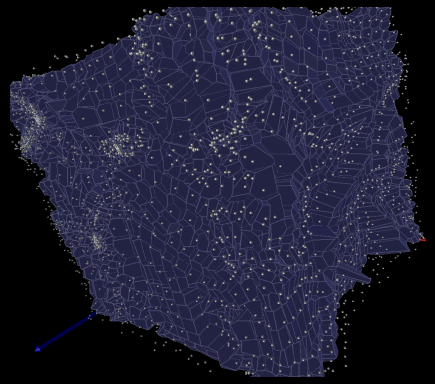
FTLE
in climate modeling



Ptychography
in materials science



Morse-Smale complex
in combustion



Voronoi and Delaunay tessellation
in cosmology

Exascale Data Analytics Software Stack

Applications

Exascale simulations, experiments, observations, ensembles

User Libraries and Tools

Analysis libraries, standard visualization/analysis packages

Data Movement

DIY	**Decaf**
(block parallelism)	(decoupled dataflows)

Data movement building blocks within one component (DIY) and between components (Decaf)

System Libraries

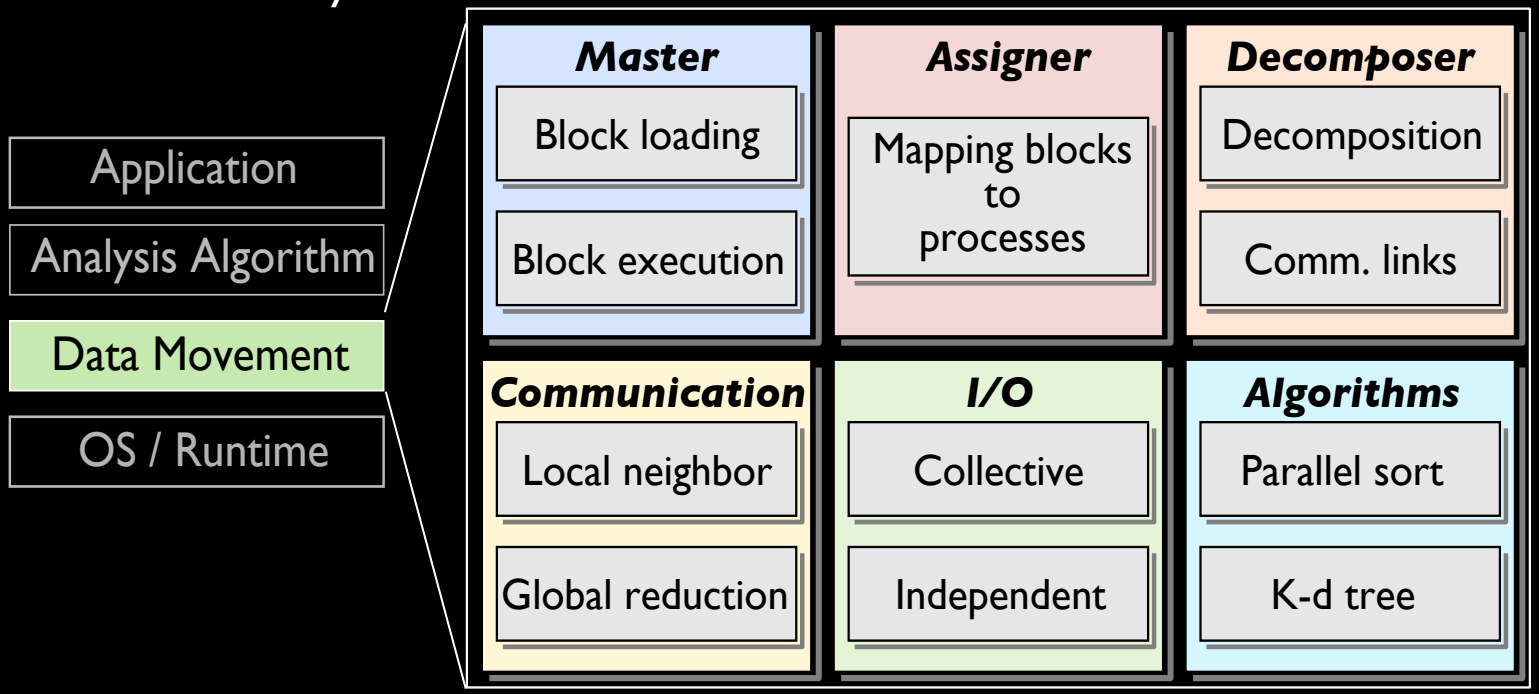
Programming model and runtime

System Services

Storage systems, resource managers, schedulers

Tom Peterka, ANL
Dmitriy Morozov, LBNL
github.com/diatomic/diy2

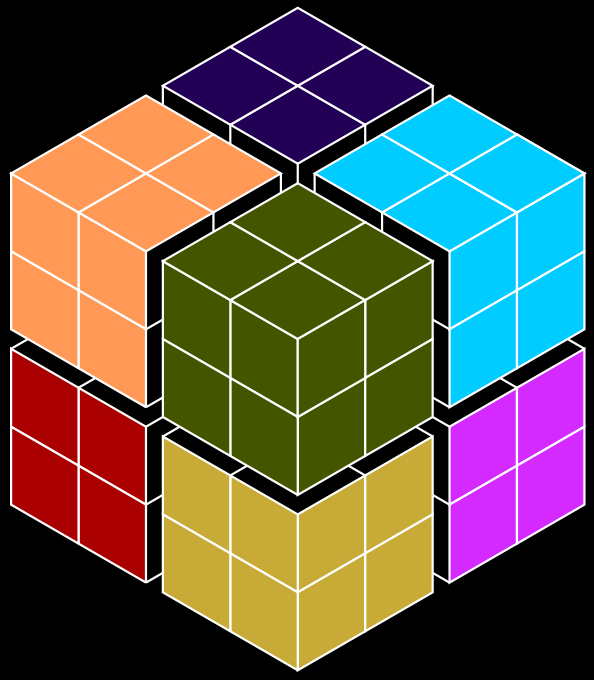
DIY



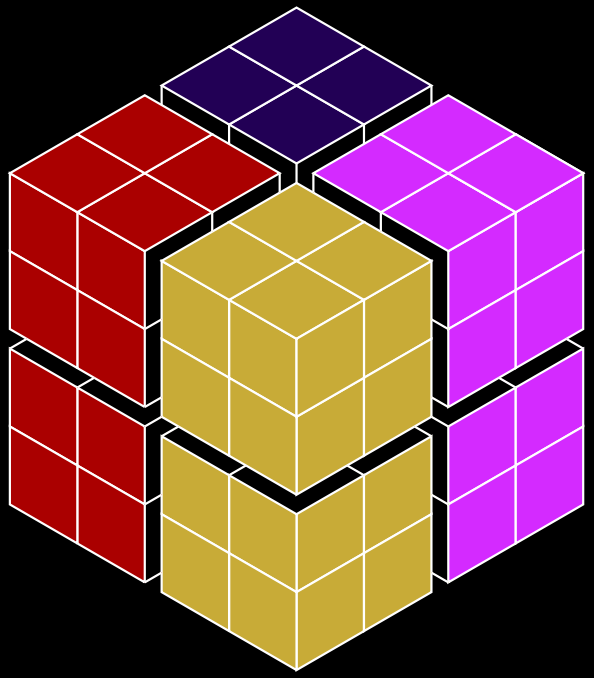
DIY is a programming model and runtime for HPC block-parallel analytics.

- Block parallelism
- Flexible domain decomposition and assignment to resources
- Efficient reusable communication patterns
- Automatic dual in- and out-of-core execution
- Automatic block threading

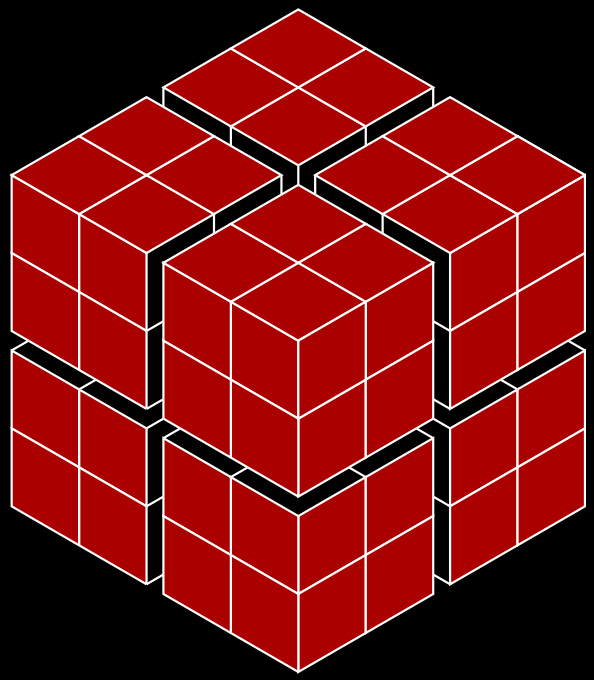
Block Parallelism



8 processes



4 processes



1 process

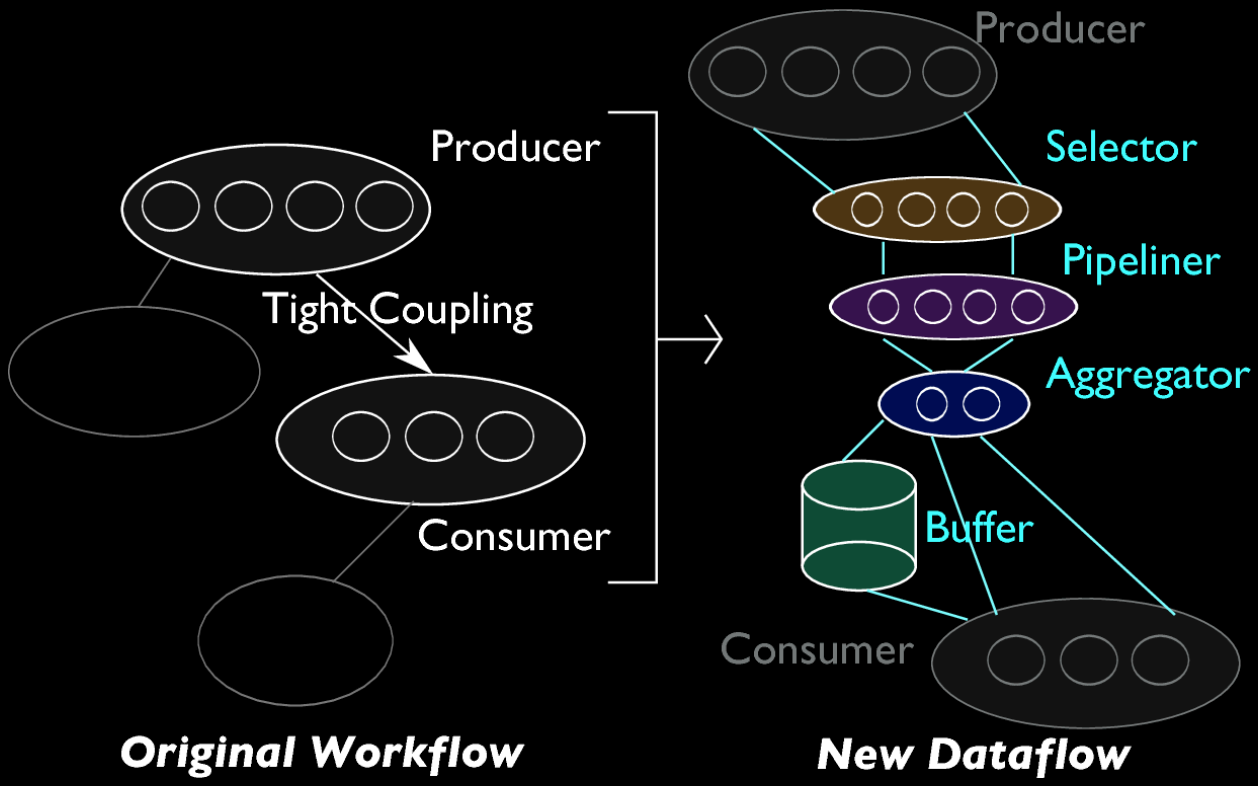
Blocks are units of work and communication; blocks exchange information with each other using DIY's communication algorithms. DIY manages block placement in MPI processes and memory/storage. This allows for flexible, high performance programs that are easy to write and debug.

```
// initialization
Master                  master(world, num_threads, mem_blocks, ...);
ContiguousAssigner      assigner(world.size(), tot_blocks);
decompose(dim, world.rank(), domain, assigner, master);

// compute, neighbor exchange
master.foreach(&foo);
master.exchange();

// reduction
RegularSwapPartners(dim, tot_blocks, k);
reduce(master, assigner, partners, &foo);

// callback function for each block
void foo(void* b, const Proxy& cp, void* aux)
{
    for (size_t i = 0; i < in.size(); i++)
        cp.dequeue(cp.link()->target(i), incoming_data);
    // do work on incoming data
    for (size_t i = 0; i < out.size(); i++)
        cp.enqueue(cp.link()->target(i), outgoing_data[i]);
}
```

Example Usage

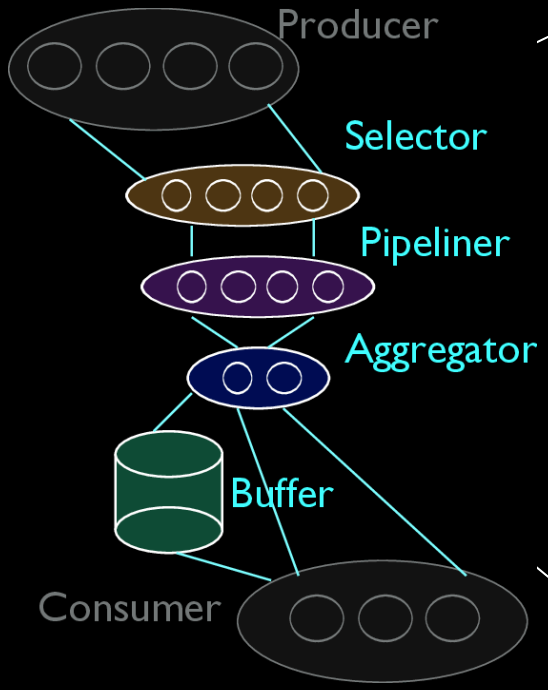
Decaf: Data Movement Between Components

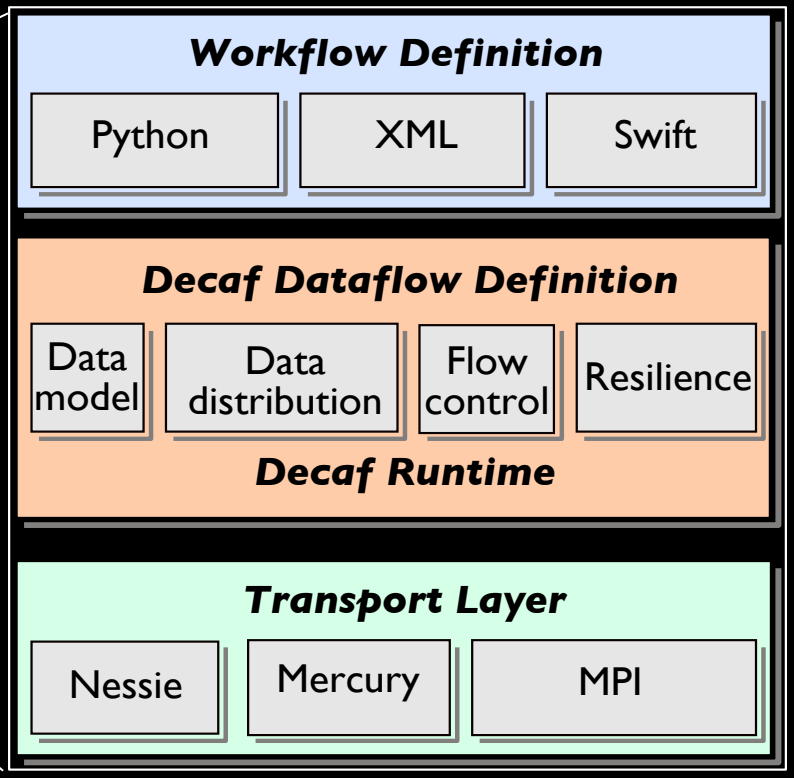
Decoupling by converting a single link into a dataflow enables new features such as fault tolerance and improved performance.



Tom Peterka,
Franck Cappello, ANL
Jay Lofstead, SNL
bitbucket.org/tpeterka1/decaf

Decaf





Decaf is a programming model and runtime for coupling HPC codes.

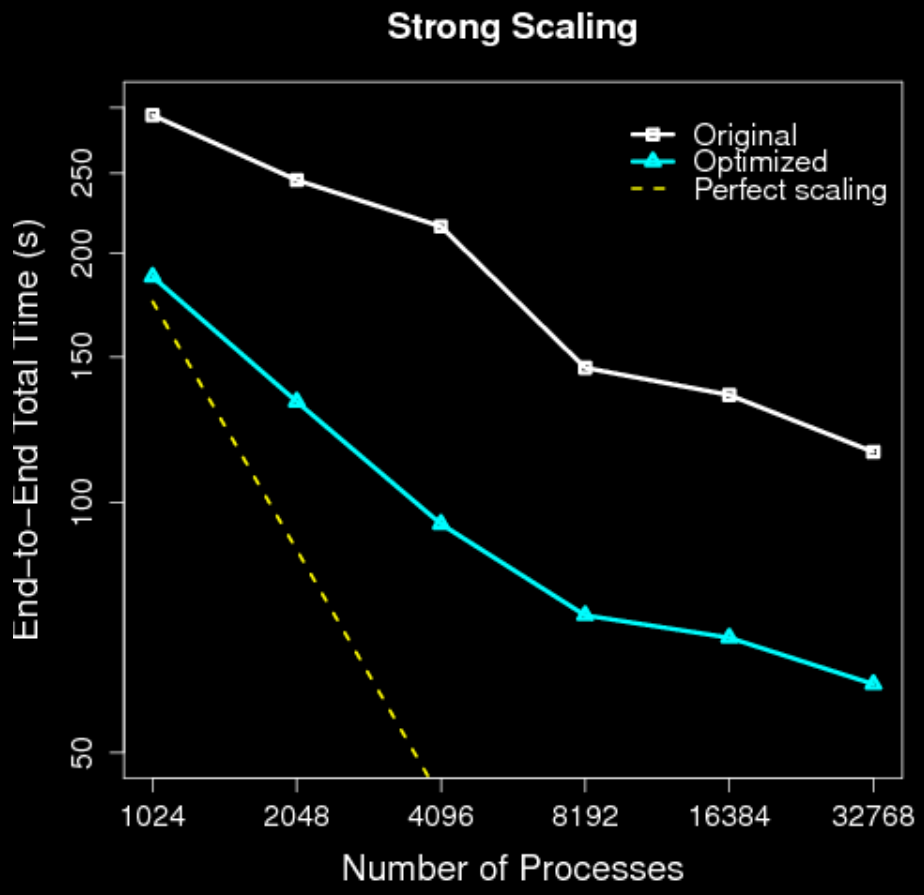
- Decoupled workflow links with configurable dataflow
- Data redistribution patterns
- Flow control
- Resilience

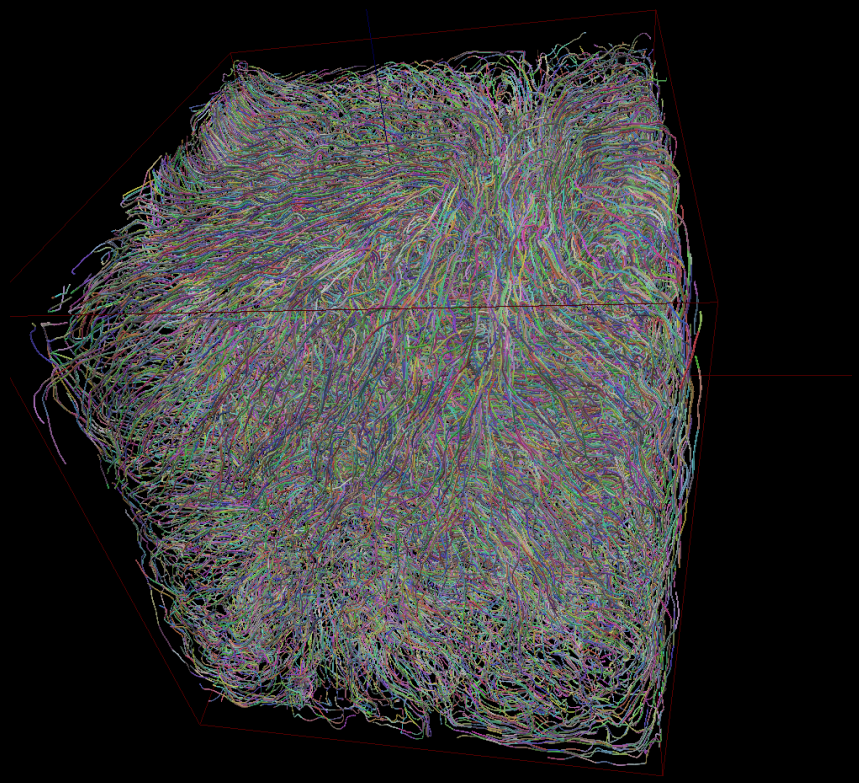
Performance Matters

With Paul Fischer, Aleks Obabko (ANL), Daniel Livescu, Mark Peterson (LANL), Jackie Chen, Ray Grout (SNL)

[github.com/GRAVITYLab/OSUFlow]

Particle Tracing in Nuclear Engineering, Mixing, Combustion



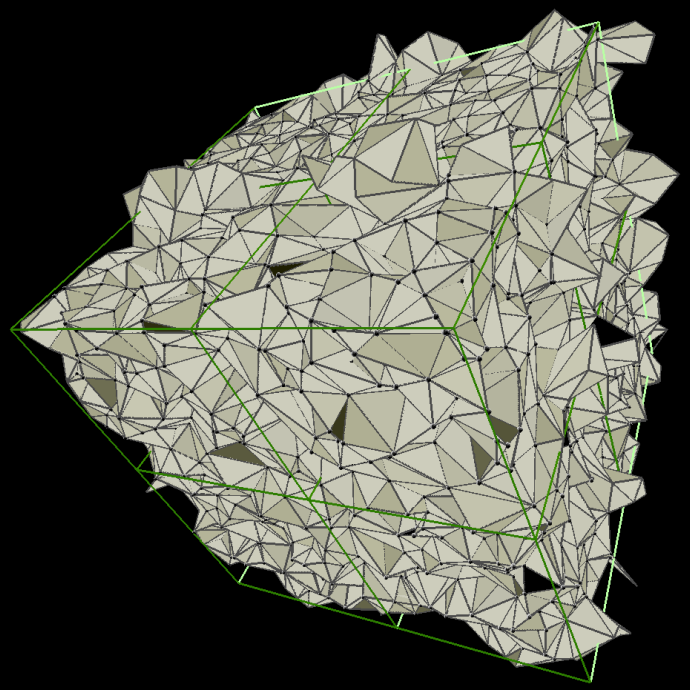


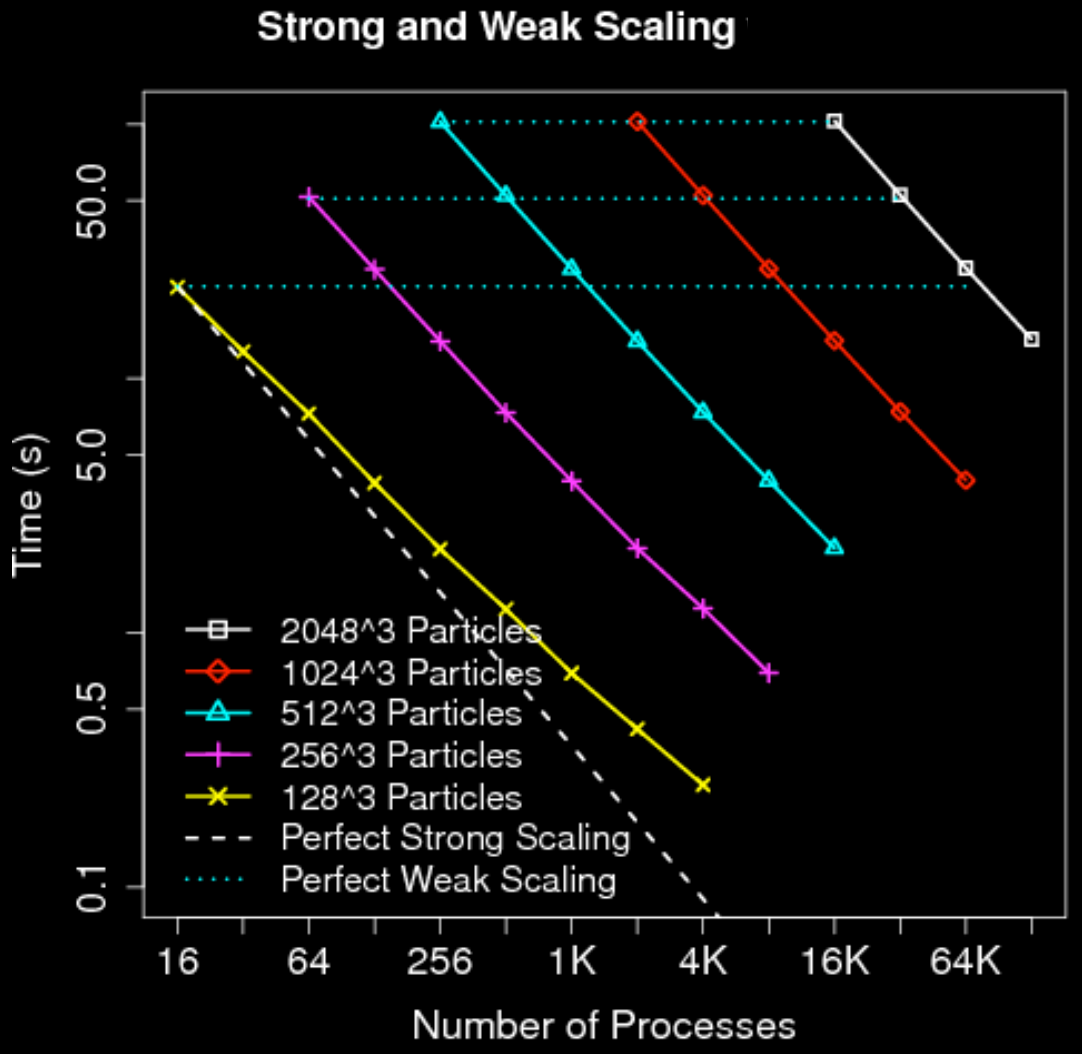
Particle tracing of ¼ million particles in a 2048^3 thermal hydraulics dataset results in strong scaling to 32K processes and an overall improvement of 2X over earlier results.

Peterka et al., A Study of Parallel Particle Tracing for Steady-State and Time-Varying Flow Fields. IPDPS '11.

[github.com/diatomic/tess2]

Computational Geometry in Cosmology and Molecular Dynamics



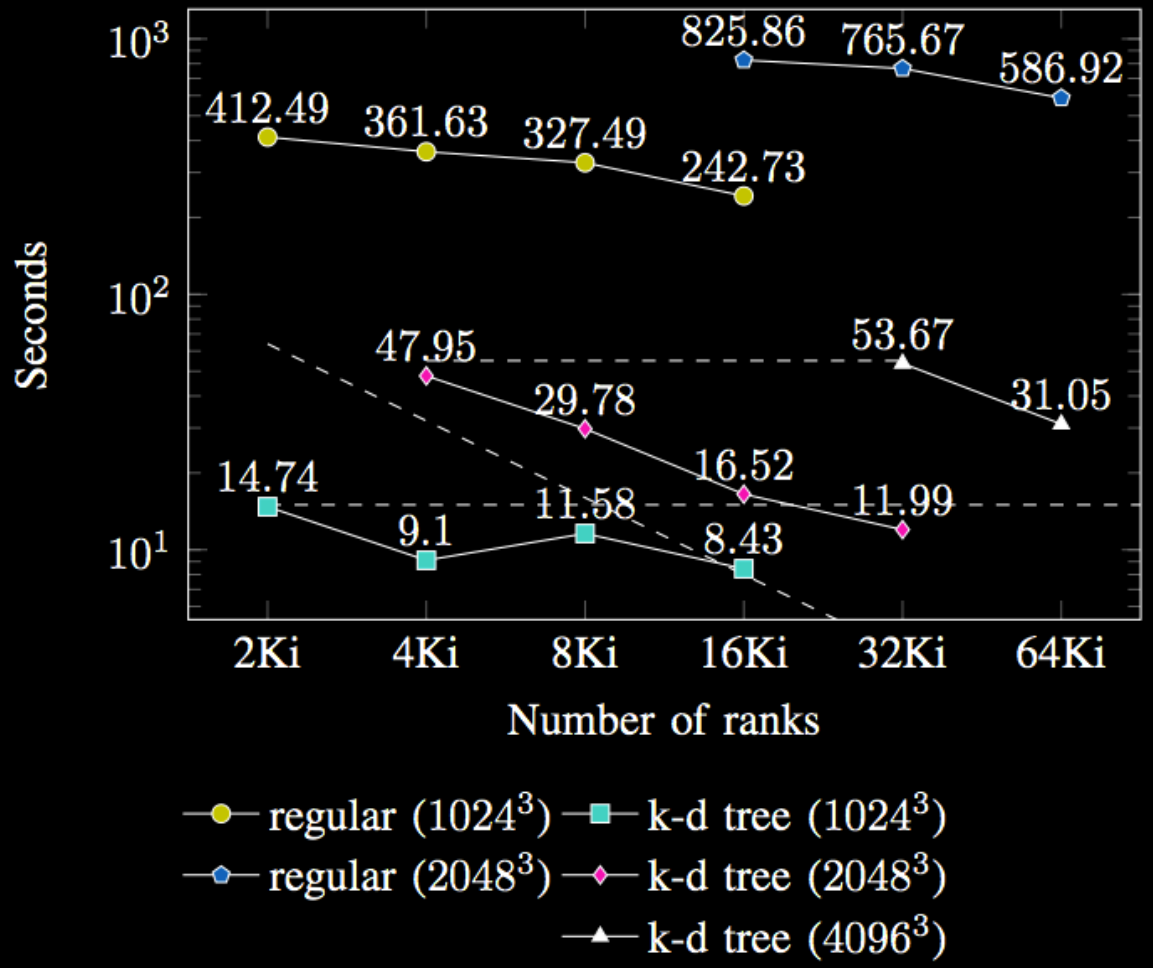


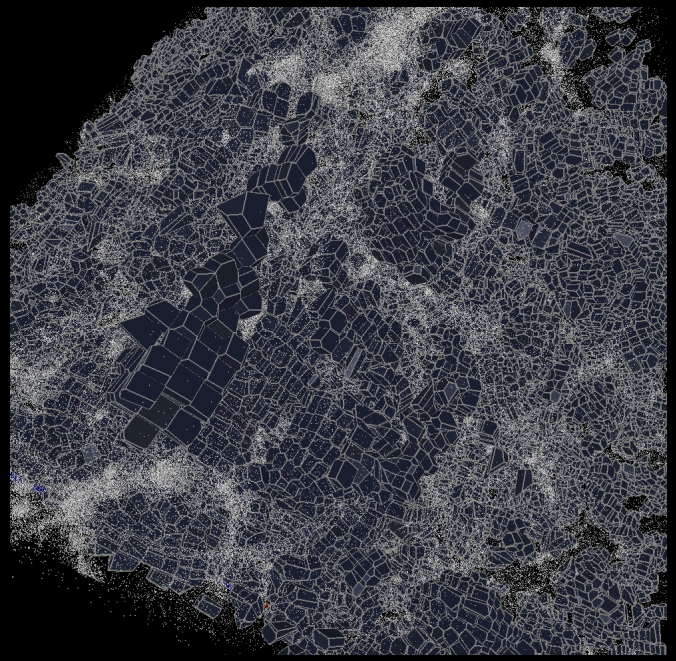
Strong and weak scaling for up to 2048^3 synthetic particles and up to 128K processes (excluding I/O) shows up to 90% strong scaling and up to 98% weak scaling.

Courtesy Dmitriy Morozov with Zarija Lukic

[github.com/diatomic/tess2]

Load Balancing in Cosmology



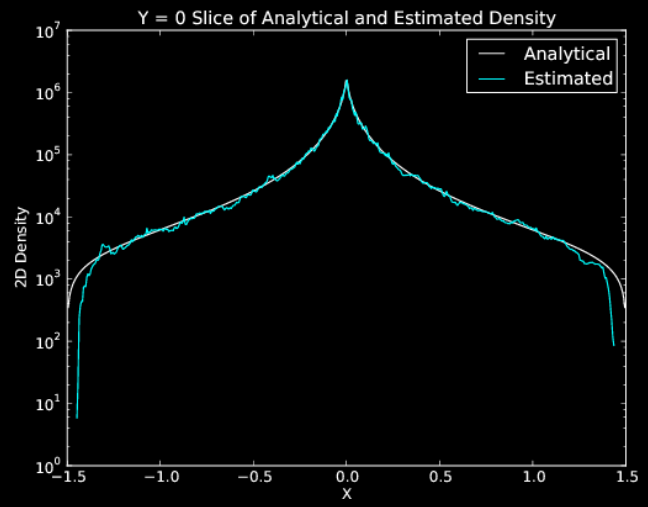


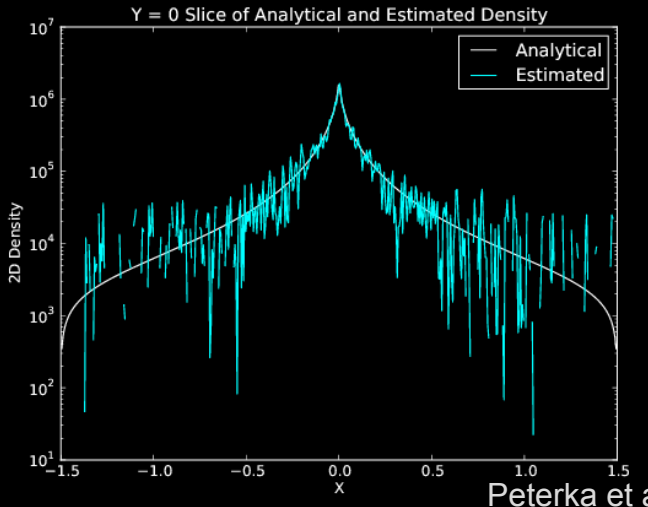
Cosmology simulations have severe load imbalance. Tessellating meshes using a k-d tree instead of regular grid results in dramatically improved performance.

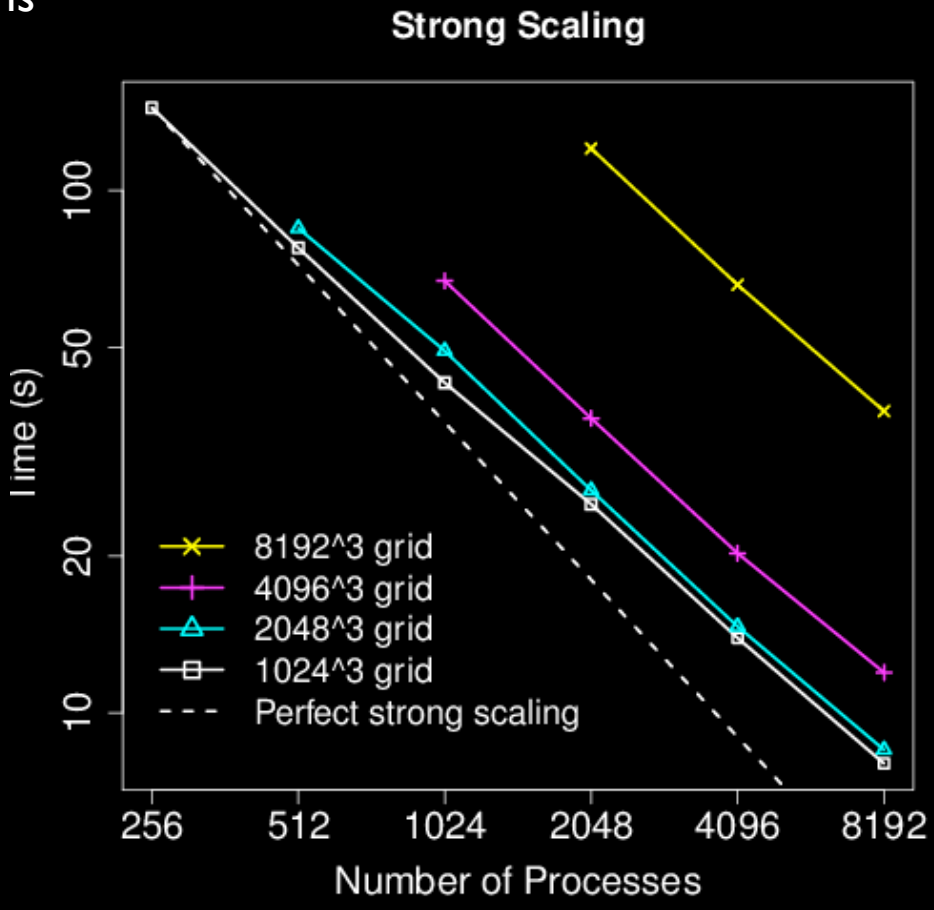
Morozov and Peterka, Efficient Delaunay Tessellation Through K-D Tree Decomposition, in preparation.

Density Estimation in Cosmology

Tessellation-based density estimation is parameter free, shape free, and automatically adaptive







Above: Strong scaling of estimating the density of 512^3 synthetic particles onto grids of various sizes.

Left: comparison of tessellation-based and CIC density

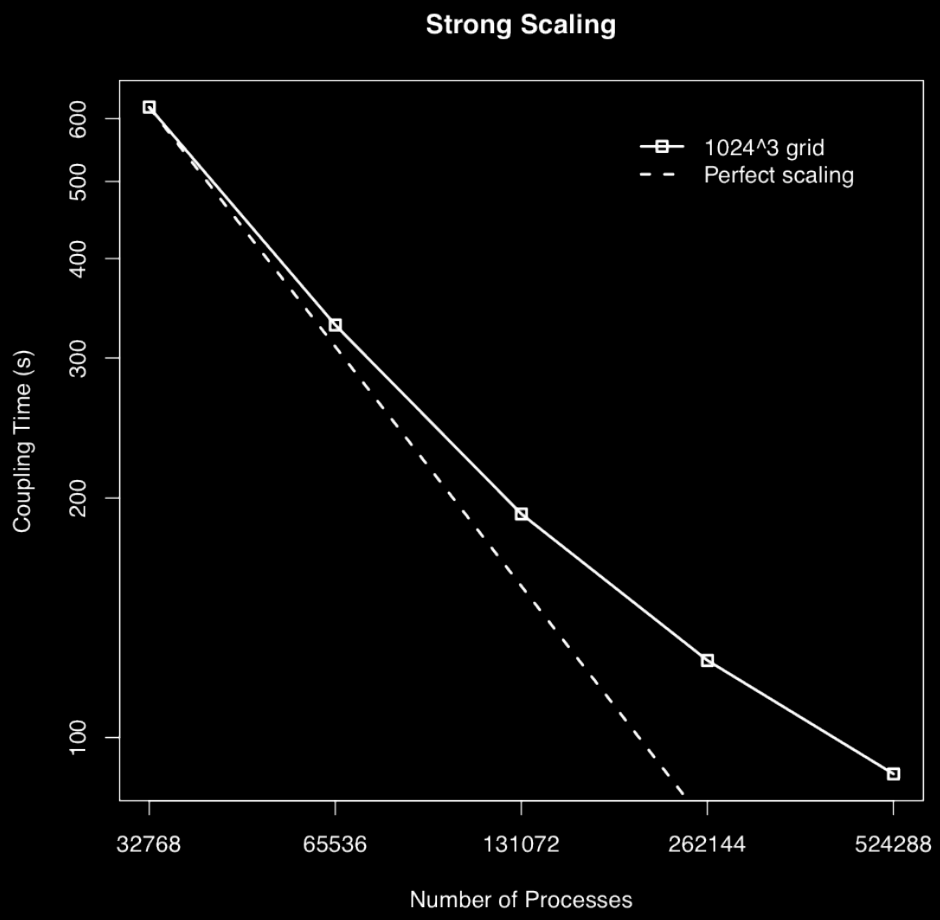
Peterka et al., Self-Adaptive Density Estimation, SIAM SISC 2016.

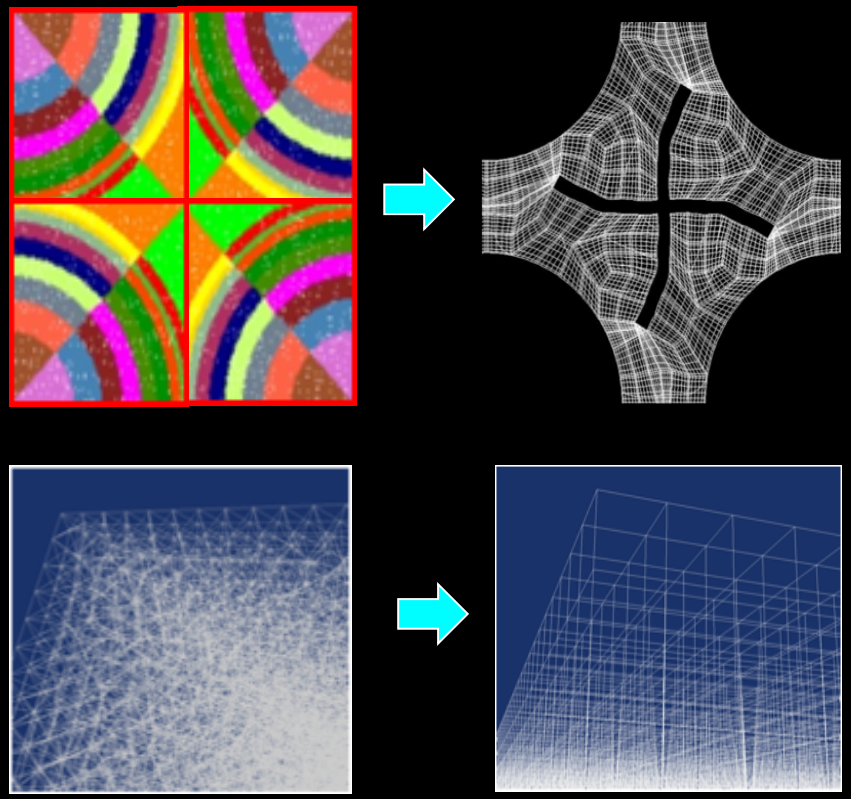
With Vijay Mahadevan, Iulian Grindeanu, Tim Tautges, Andrew Siegel

[github.com/tpeterka/cian2]

Codesign in Nuclear Engineering

The cian proxy app of the CESAR codesign center emulate multiphysics coupling between neutronics and thermal hydraulics in nuclear reactor design.





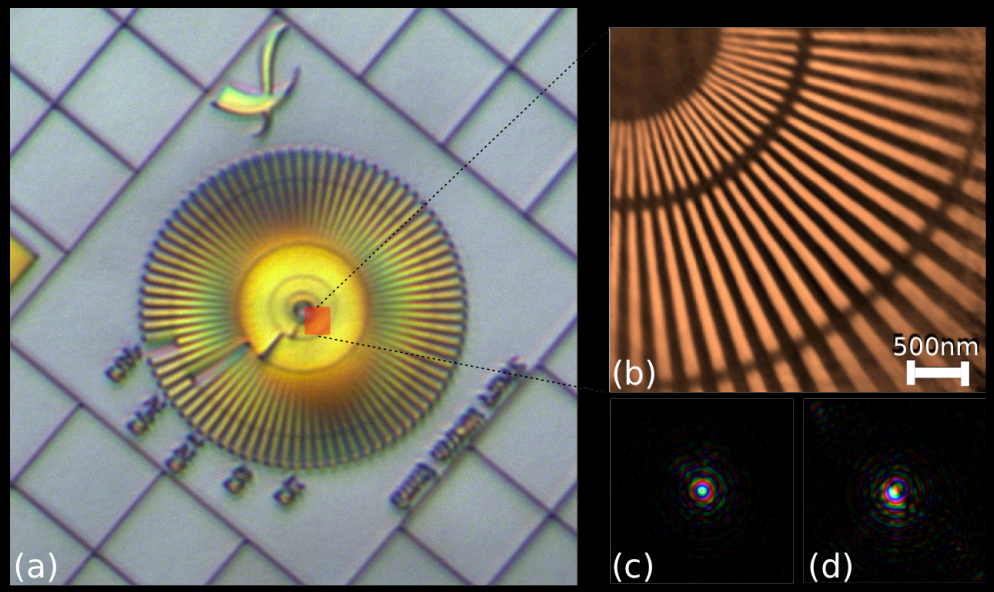
Coupling proxy app to transfer a solution from a 1024^3 tetrahedral mesh to a 1024^3 hexahedral mesh and back again at up to ½ million blocks (MPI processes) and 43% strong scaling efficiency.

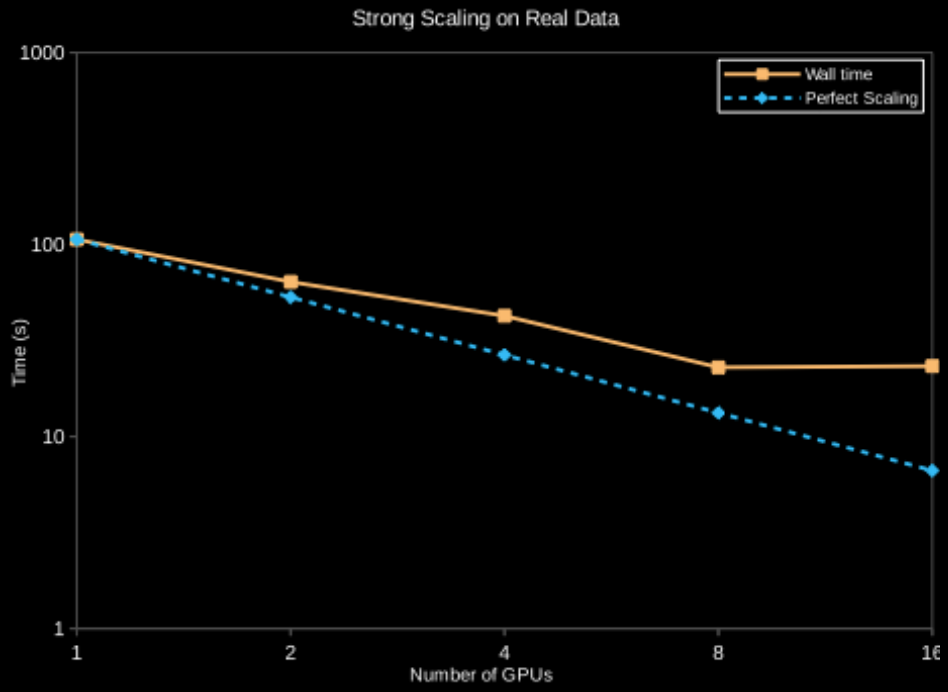
Courtesy Youssef Nashed, with David Vine, Chris Jacobsen (APS)

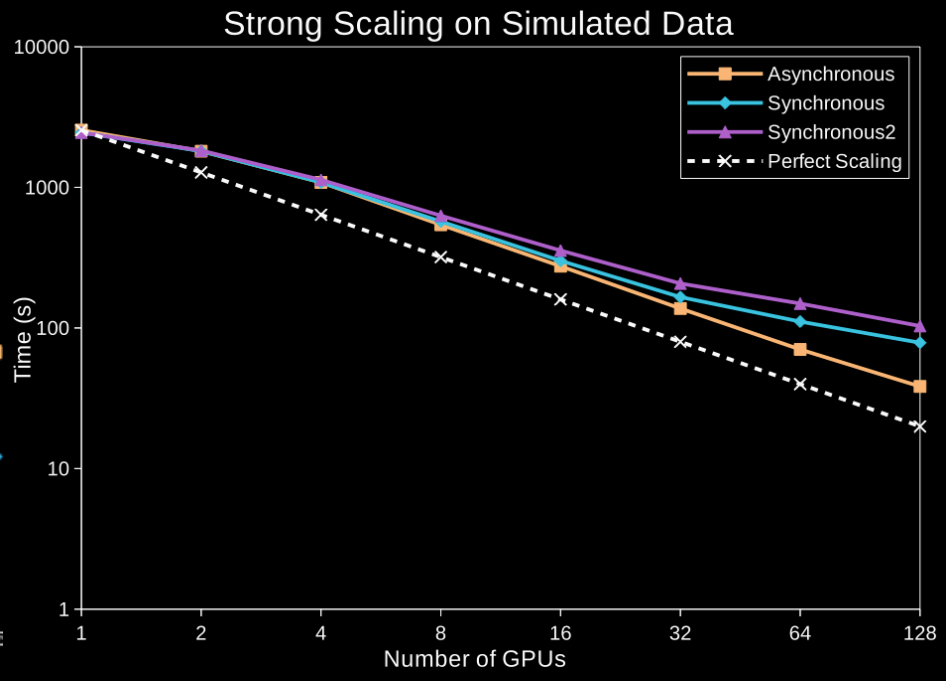
[ptycholib (available on request)]

Ptychographic Reconstruction at the APS

A test pattern, with 30 nm feature size, was raster scanned using a 5.2 keV X-ray beam. The total scanning time was about 20 minutes. Reconstruction time was ~2 minutes.







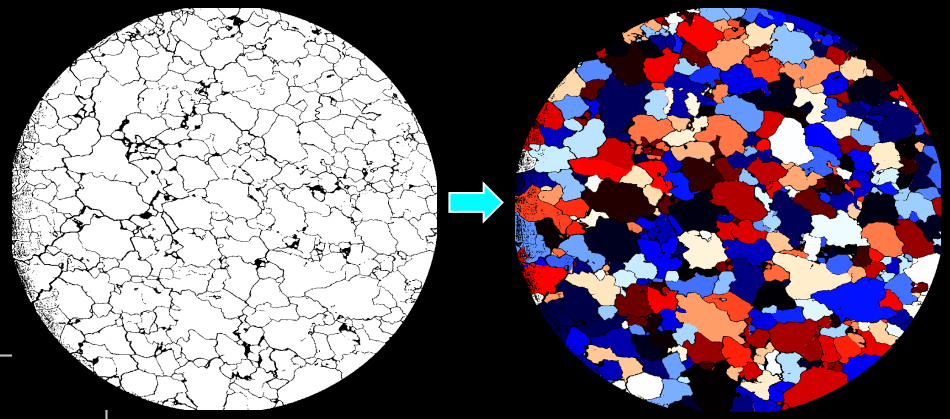
Nashed et al., Parallel Ptychographic Reconstruction. Optics Express 2014.

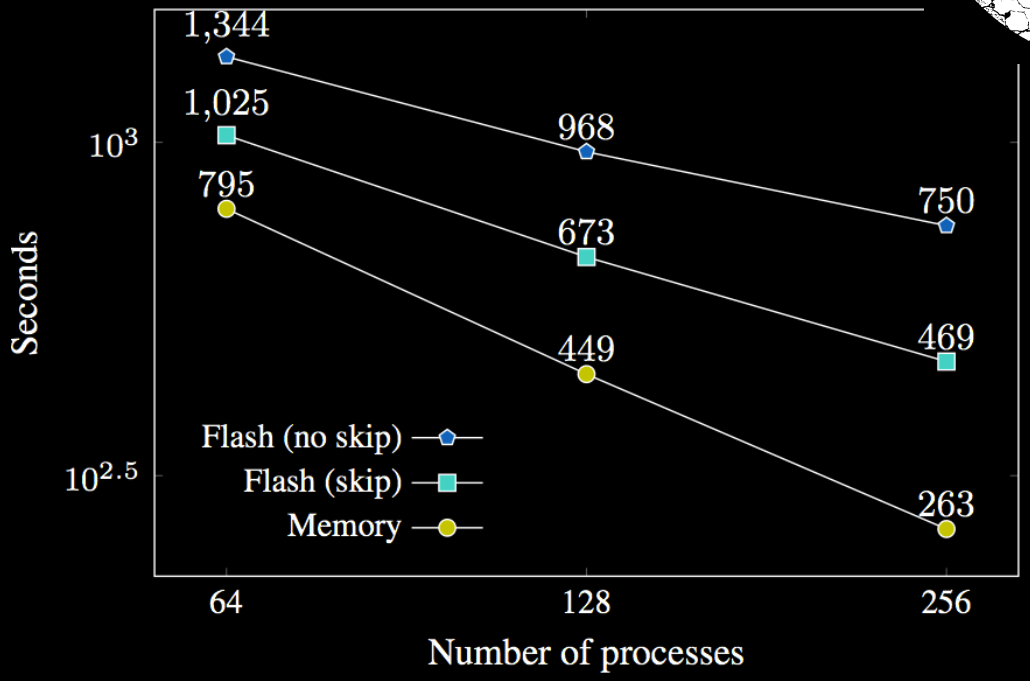
Courtesy Dmitriy Morozov and Patrick O'Neil with Michael Manga, Dula Parkinson (ALS)

[bitbucket.org/mrzv/gaia]

Image Segmentation at the ALS

Researchers at LBL developed tools for segmentation and connectivity analysis of granular and porous media using DIY2.





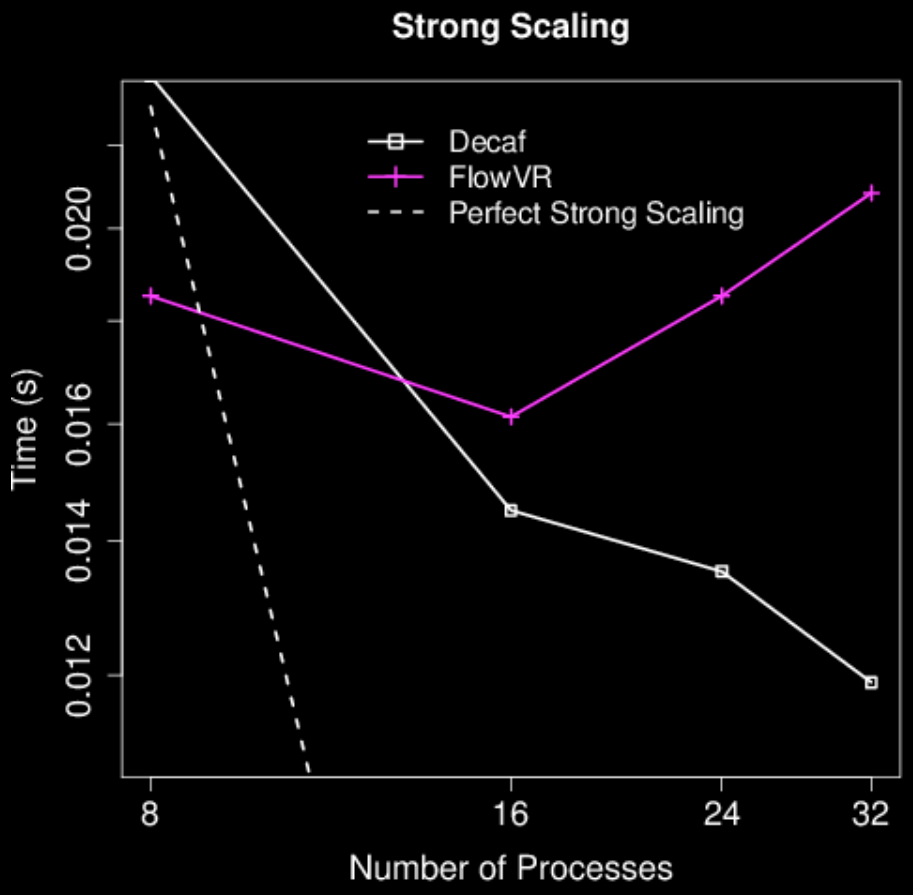
Left: 3D image of a granular material (flexible sandstone) acquired at ALS by Michael Manga and Dula Parkinson. (Data: 2560 × 2560 × 1276). Right: Watershed segmentation of the material identifies individual grains (run on Edison @ NERSC) [courtesy Morozov, O'Neil (LBL)].

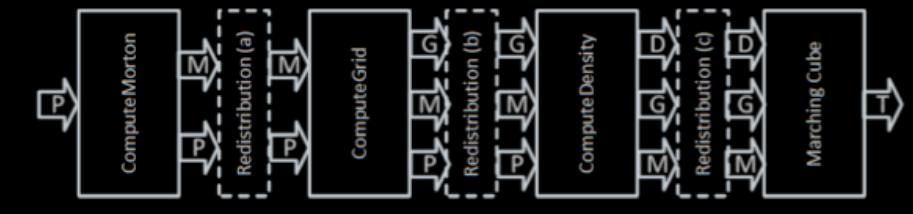
Morozov and Peterka, Block-Parallel Data Analysis with DIY2. Submitted to SPAA 2016.

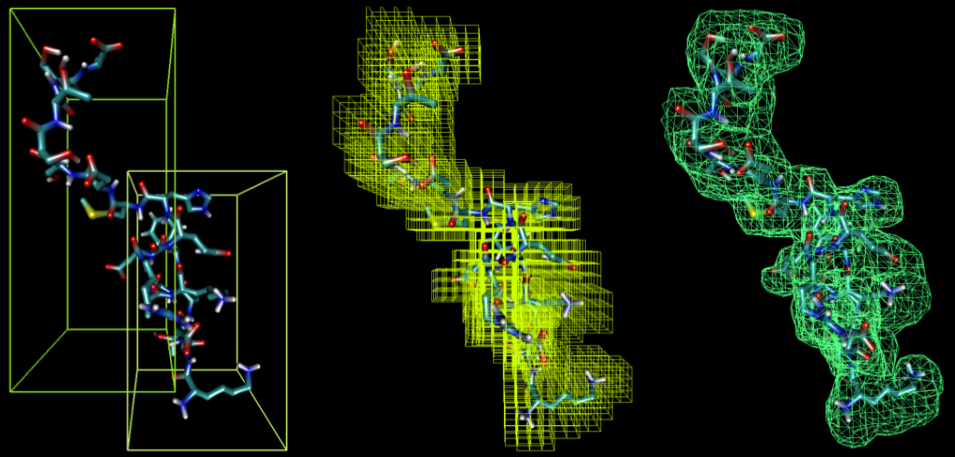
Courtesy Matthieu Dreher

[bitbucket.org/tpeterka1/decaf]

Data Redistribution in Molecular Dynamics





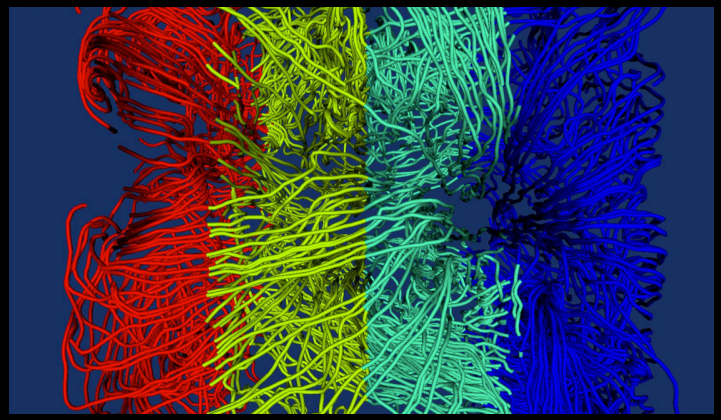


Three different redistributions are performed while computing an isosurface from an MD simulation of 54,000 lipids(2.1M particles). [Dreher et al. 2014]

We applied the decaf redistribution library to the Gromacs molecular dynamics code in order to visualize isosurfaces from molecular density. Code complexity was reduced dramatically, while maintaining performance improved.

Looking Ahead

DIY Integration with VTK and Friends



Thermal hydraulics vector data from Nek5000 with parallel particle tracing using DIY1/OSUFlow rendered in VTK.
[Courtesy Zhanping Liu]

- VTK
  - *At ANL and Kitware*: DIY now a 3rd party library in VTK build, used for parallel resample filter (for LANL ASC milestone)
- Paraview
  - *At Kitware*: In 2015, Kitware tested DIY's kd-tree algorithm in Paraview; production release slated for ParaView 5. in 2016 w/ DIY a 3rd party library. Parallel resample filter w/ DIY now also in ParaView.
- VisIt
  - *At ANL*: Summer student 2016 to rewrite VisIt volume renderer with DIY.
- VTK-m
  - Integration in ECP

Sewell et al., The SDAV Software Frameworks for Visualization and Analysis on Next-Generation Multi-Core and Many-Core Architectures, UltraVis '12.

DIY R&D: Exascale Architectures

Dynamic block execution/communication patterns

1. Synchronization: Relax BSP synchronization, e.g., for iterative algorithms, to overlap communication with computation.

2. Load balance: Support work stealing and other dynamic load balancing algorithms.

Memory/storage hierarchy

3. Deep hierarchy: Supporting more than 2 levels of hierarchy and scheduling block movement, including prefetching, in nearby levels.

4. Resilience: Support fault tolerance with out of core block movement.

DIY R&D: Exascale Applications

Apply new decompositions

1. *Astrophysics*: Add support for AMR (block-based and patch-based).
2. *Climate*: Use existing simulation decompositions (eg., geodesic grid).

Adaptive algorithms

3. *Combustion*: Select and vary communication pattern parameters automatically (eg., # blocks in memory).
4. *Reactors*: Include work stealing algorithms to load-balance unstructured time-varying graph / mesh partitions.
5. *Cosmology*: Incrementally adapt time-varying decomposition (eg., adapt k-d tree on the fly instead of rebuilding)

Decaf R&D & Integration

Allow cycles in workflow graphs

Develop buffering component

Deploy multiple executables

Develop more use cases

Integrate with other workflow tools

1. *Swift*
2. *ADIOS*
3. *Catalyst?*

References

DIY

- Peterka, T., Ross, R., Kendall, W., Gyulassy, A., Pascucci, V., Shen, H.-W., Lee, T.-Y., Chaudhuri, A.: Scalable Parallel Building Blocks for Custom Data Analysis. LDAV 2011.
- Peterka, T., Ross, R.: Versatile Communication Algorithms for Data Analysis. IMUDI 2012.
- Sewell, C., Meredith, J., Moreland, K., Peterka, T., DeMarle, D., Lo, Li-ta, Ahrens, J., Maynard, R., Geveci, B.: The SDAV Software Frameworks for Visualization and Analysis on Next-Generation Multi-Core and Many-Core Architectures. UltraVis 2012.
- Morozov, D., Peterka, T.: Block-Parallel Data Analysis with DIY2. In preparation, 2016.

https://github.com/diatomic/diy2

Decaf

- Wozniak, J., Peterka, T., Armstrong, T., Dinan, J., Lusk, E., Wilde, M., Foster, I.: Dataflow Coordination of Data-Parallel Tasks via MPI 3.0. EuroMPI, 2013.
- Dorier, M., Dreher, M., Peterka, T., Wozniak, J., Antoniu, G., Raffin, B.: Lessons Learned from Building In Situ Coupling Frameworks. Proceedings of ISAV 2015.
- Peterka, T., Croubois, H., Li, N., Rangel, E., Cappello, F.: Self-Adaptive Density Estimation of Particle Data. To appear SIAM SISC 2016.

https://bitbucket.org/tpeterka1/decaf

DIY applications

- Peterka, T., Morozov, D., Phillips, C.: High-Performance Computation of Distributed-Memory Parallel 3D Voronoi and Delaunay Tessellation. SC14.
- Lu, K., Shen, H.-W., Peterka, T.: Scalable Computation of Stream Surfaces on Large Scale Vector Fields. SC14.
- Chaudhuri, A., Lee-T.-Y., Shen, H.-W., Peterka, T.: Efficient Indexing and Querying of Distributions for Visualizing Large-scale Data. Proceedings of IEEE PacificVis 2014.
- Nashed, Y., Vine, D., Peterka, T., Deng, J., Ross, R., Jacobsen, C.: Parallel Ptychographic Reconstruction. Optics Express 2014.
- Gyulassy, A., Peterka, T., Pascucci, V., Ross, R.: The Parallel Computation of Morse-Smale Complexes. IPDPS 2012.
- Nouanesengsy, B., Lee, T.-Y., Lu, K., Shen, H.-W., Peterka, T.: Parallel Particle Advection and FTLE Computation for Time-Varying Flow Fields. SC12.
- Chaudhuri, A., Lee-T.-Y., Zhou, B., Wang, C., Xu, T., Shen, H.-W., Peterka, T., Chiang, Y.-J.: Scalable Computation of Distributions from Large Scale Data Sets. LDAV 2012.
- Peterka, T., Ross, R., Nouanesengsy, B., Lee, T.-Y., Shen, H.-W., Kendall, W., Huang, J.: A Study of Parallel Particle Tracing for Steady-State and Time-Varying Flow Fields. IPDPS 2011.
- Peterka, T., Kendall, W., Goodell, D., Nouanesengsy, B., Shen, H.-W., Huang, J., Moreland, K., Thakur, R., Ross, R.: Performance of Communication Patterns for Extreme-Scale Analysis and Visualization. SciDAC 2010.

U.S. DEPARTMENT OF ENERGY

Argonne NATIONAL LABORATORY

Acknowledgments

Facilities

Argonne Leadership Computing Facility (ALCF)

Oak Ridge National Center for Computational Sciences (NCCS)

National Energy Research Scientific Computing Center (NERSC)

Funding

DOE SDMAV Exascale Initiative

DOE SciDAC SDAV Institute

People

DIY: Dmitriy Morozov (LBNL)

Decaf: Franck Cappello (ANL), Jay Lofstead (SNL)

https://github.com/diatomic/diy2

https://bitbucket.org/tpeterka1/decaf

Tom Peterka

tpeterka@mcs.anl.gov

http://www.mcs.anl.gov/~tpeterka

Mathematics and Computer Science Division